[0045]

VIDEO CARD CLASS - TABULATED VIEW
CLASS NAME: VideoCard

| ATTRIBUTES | Type | Comments |
|---|---|---|
| video_memory | Enum | Choice: '1 MB', '2 MB', '4 MB' |
| pixel clock | integer | |
| display | Monitor | |

[0046]

VIDEO CARD CLASS - PSEUDO-CODE

```
class VideoCard; # define the video card class
{
  attributes: # define the attributes required by the VideoCard class
  {
      video_memory: # define the characteristics of the attribute
                    # video_memory
        {
          type : enum ; # it's of type enum
                      # having a choice as defined ...
          choice : ('1MB', '2MB', '4MB' );
        }
      pixel_clock:
      {
          type: integer;
      {
      display:
      {
          type: Monitor; # the display of a type Monitor
      }
  }
}
```

[0047]

MONITOR CLASS - TABULATED VIEW
CLASS NAME: Monitor

| ATTRIBUTES | Type | Comments |
|---|---|---|
| screen_resolution | Integer | |
| refresh_rate | Integer | |

[0048]

MONITOR CLASS - PSEUDO CODE

```
class Monitor; # define the monitor class
{
 attributes: # define the attributes required by the monitor class
 {
      screen_resolution:   # define the characteristics of the attribute
                           # screen_resolution
       {
            type : integer;
       }
      refresh_rate: # define the characteristics of the variable
                    # refresh_rate
      {
          type: enum; # its an enumerated type
```

-continued

MONITOR CLASS - PSEUDO CODE

```
          choice: ('50Hz', '60Hz', '75Hz');
      }
 }
}
```

[0049]

PROCESSOR CLASS - TABULATED VIEW
CLASS NAME: Processor

| ATTRIBUTES | Type: | |
|---|---|---|
| clock_frequency | integer | min = 50, max = 200 |
| cache_size | enum | Choice: '256 b', '512 b', '1024 b' |

[0050]

PROCESSOR CLASS - PSEUDO CODE

```
class Processor; # define the processor class
{
 attributes: # define the attributes required by the video card class
 {
      clock_frequency: # define the characteristics of the attribute
                       # clock_frequency
       {
          type : integer; # it's of type integer
          min = 50 ; # set the minimum value acceptable
          max = 200 ; # set the maximum value acceptable
       }
      cache_size: # define the characteristics of the attribute cache_size
      {
          type: enum; # its an enumerated type
          choice: ('256b', '512b', '1024b');
      }
 }
}
```

[0051] In a Perl-type implementation the class implementations defined using the declarative class definitions above are created at run-time by the class-making module. A main program creates the configuration model in memory by creating instances (or objects) of the class implementations. The main program may, for example, implement a user interface allowing users to read, set or modify attributes of each object making up the configuration model. It will be appreciated that the main program may also use classes defined in a more conventional, non-declarative, manner.

[0052] As previously described, the use of generic class-making modules along with declarative class definitions has currently been limited for use where there are no dependencies between nodes or leaves of a configuration tree. The present invention helps to overcome such disadvantages by providing a framework through which dependency information may defined in a declarative manner and thereby be used with a generic class-making module.

[0053] One of the problems of using generic class-making modules, such as Class::MethodMaker, is that referencing of objects arranged in a hierarchical tree-like structure becomes